1 Understanding Mapping Reductions

1.1 Recap

Definition 1. Language $A$ is mapping reducible to language $B$ ($A \leq_m B$) if there is a computable function $f: \Sigma^* \rightarrow \Sigma^*$, where for every $w$, $w \in A \iff f(w) \in B$

Example 2. Let’s consider the following languages $A$, $B$.

$A = \{ \text{strings in } \{a, b\}^* \text{ with amount of a’s = amount of b’s} \}$

$B = \{a^n b^n : n \geq 0 \}$.

We can construct our mapping $f$ to be, for example,

$f(w)$: on input $w$, sorting $w$ by the symbols (and thus putting all a’s before b’s).

- for any string $w$, $f(w) \in B$ if and only if $w \in A$
- note that this example is onto (thought it doesn’t have to be), but not one-to-one

1.2 Mapping Reduction in relation to Turing Reduction

Theorem 3. If $A \leq_m B$, then $A \leq_T B$

Proof. Let $f$ be the mapping, and $D_B$ the decider for $L(B)$.

Then, we can construct a decider $D_A$ for $A$ on input $w$:

$D_A$ on input $w$:

- compute $y = f(w)$
- run $D_B(y)$ and output the same

Fact 4. On the other hand, if $A \leq_T B$, then this does not necessarily mean that $A \leq_m B$. However, it does in the following special case: check if the Turing reduction uses the Decider once, and always outputs same (no flipping of accept/reject). In this case, the Turing reduction implies Mapping reduction.

Theorem 5. If $A \leq_T B$, and this reduction calls the decider for $B$, $D_B$ exactly once and outputs same. Then, $A \leq_M B$. 
Proof. Let $A \leq_T B$ via a reduction that constructs a decider $D_A$ for $A$, by calling a decider $D_B$ for $B$ exactly once and outputting the same thing as $D_B$. We define $f(w)$ as the input that $D_B$ is invoked on when $D_A$ starts with input $w$. This $f$ is clearly computable, as this is exactly what $D_A$ computes on input $w$, before calling $D_B$ on $f(w)$. It satisfies that $w$ is in $A$ if and only if $f(w)$ is in $B$, because of the fact that the output of $D_A$ on $w$ is the same as the output of $D_B$ on $f(w)$.

1.3 Mapping Reduction Properties

Theorem 6. If $A \leq_m B$ and $B$ is recognizable, then $A$ is also recognizable.

Proof. Let $f$ be the mapping, and $M_B$ the recognizer for $B$.

We can construct a recognizer $M_A$ on input $w$: $M_A$ on input $w$:

- compute $y = f(w)$
  - we are using the fact that $f$ is a computable function (as part of the def. of mapping reduction)
- run $M_B(y)$
- if it accepts, accept.
- if it rejects, reject.

Analysis: Note that since $f$ is a computable function, the first step is computable in finite time.

if $w \in A$

$\Rightarrow y = f(w) \in B$

$\Rightarrow M_B(y)$ accepts

$\Rightarrow M_A(w)$ accepts

if $w \notin A$

$\Rightarrow y = f(w) \notin B$

$\Rightarrow M_B(y)$ rejects or runs forever

$\Rightarrow M_A(w)$ rejects or loops forever

Theorem 7. If $A \leq_m B$ iff $\overline{A} \leq_m \overline{B}$.

Proof. Prove both ways for equivalence:

$\rightarrow$: If $A \leq_m B$, then $\overline{A} \leq_m \overline{B}$

Let’s assume $A \leq_m B$, and the mapping for this reduction to be $f$. We know that our mapping $f$ satisfies $w \in A \leftrightarrow f(w) \in B$. This is equivalent to $w \notin A \leftrightarrow f(w) \notin B$ for all mapping (will always answer yes/yes, no/no). Thus we can use this same $f$ to map $\overline{A} \leq_m \overline{B}$.

$\leftarrow$: If $\overline{A} \leq_m \overline{B}$, then $A \leq_m B$: Follows from the above, by noticing that $\overline{A} = A$ and $\overline{B} = B$. 

2
Some other possible exercises:

**Corollary 8.** If $A \leq_m B$ and $A$ is not recognizable, then $B$ is not recognizable.

*Proof.* Same reasoning with same $f$ construction (understand it as contrapositive of Theorem 7). □

**Theorem 9.** If $A \leq_m B$ and $B$ is decidable, then $A$ is also decidable.

**Corollary 10.** If $A \leq_m B$ and $A$ is not decidable, then $B$ is not decidable.

**Corollary 11.** If $A \leq_m B$, and $\overline{A}$ is not recognizable ($A$ is not co-recognizable), then $B$ is not recognizable.

**Fact 12.** Note that for Turing reductions you can add complements arbitrarily. For mapping reduction you can only add complement to both sides, not just one at a time – otherwise it may no longer be true.

**Example 13.** $EQ_{TM} = \{\langle M_1, M_2 \rangle : M_1, M_2 $ are TMs and $L(M_1) = L(M_2)\}$ is neither recognizable nor co-recognizable.

**Claim 14.** $EQ_{TM}$ is not recognizable.

*Proof.* By the previous lecture, we have seen $E_{TM} \leq_T EQ_{TM}$. This reduction is actually a mapping reduction where our $f$ (mapping) corresponds to $f(\langle M \rangle)$: $\langle M_\emptyset, M \rangle$, where $M_\emptyset$ is the TM that rejects all inputs (This is a sufficient explanation because we already know $E_{TM}$ is not recognizable).

This is mapping reduction because: $f$ is a computable function, since it involves outputting the encoding of $M_\emptyset$ (which can be hard coded into our algorithm), followed by the encoding $M$, which is just copying of the input.

$$\langle M \rangle \in E_{TM}:$$
$$\Leftrightarrow M \text{ is a TM and } L(M) = \emptyset = L(M_\emptyset)$$
$$\Leftrightarrow \langle M_\emptyset, M \rangle \in EQ_{TM}$$ □

**Claim 15.** $EQ_{TM}$ is not co-recognizable. (i.e. $\overline{EQ_{TM}}$ is not recognizable)

*Proof.* Let’s prove that $\overline{A_{TM}} \leq_m \overline{EQ_{TM}}$ (or equivalently, $A_{TM} \leq_m EQ_{TM}$). This is sufficient because we know that $\overline{A_{TM}}$ is not recognizable (we proved in class that $A_{TM}$ is non-decidable but recognizable).

We can construct an $f$ such that: $f(\langle M, w \rangle) = \langle M_1, M_2 \rangle$ where $M_1$ accepts all inputs, and $M_2$ for any input $x$, runs $M$ on $w$, and if it accepts, accept $x$. We can easily deduct that this $f$ is computable as well.

**Analysis**
- If $\langle M, w \rangle \in A_{TM}$
  - then $M$ accepts $w$, so $M_2$ will accepts all inputs $x$, namely $L(M_2) = \Sigma^* = L(M_1)$.
  - therefore, $\langle M_1, M_2 \rangle \in EQ_{TM}$

- If $\langle M, w \rangle \notin A_{TM}$
  - then $M$ does not accept $w$, so $M_2$ does not accept any input $x$, namely $L(M_2) = \emptyset \neq L(M_2)$.
  - therefore, $\langle M_1, M_2 \rangle \notin EQ_{TM}$ □
Example 16. Revisiting the proof of Rice’s Theorem

By complementing both sides and recalling that $\overline{A_{TM}}$ is not recognizable, we get the following revised version of the Rice Theorem. For proving the Rice Theorem, we proved that for any non-trivial language property $P$, if $\emptyset$ does not satisfy the property then we showed a mapping reduction from $A_{TM} \leq_m P$, and if $\emptyset$ does satisfy the property, then it was a mapping reduction from $A_{TM} \leq_m \overline{P}$.

Theorem 17. Rice’s Theorem
For any $P$, a non-trivial recognizable language property, if $\emptyset$ satisfies $P$ then $P$ is not recognizable, and if $\emptyset$ does not satisfy $P$ then $\overline{P}$ is not recognizable. In either case, $P$ is undecidable.

Example 18. $CLT_{TM} = \{\langle M \rangle :$ where $M$ is a TM and $L(M)$ is a context free language$\}$ Deduct that $CLT_{TM}$ is non recognizable with the Refined Version of the Rice’s Theorem.

Proof. We show that $CLT_{TM}$ is a non-trivial property of TM languages, and that $\emptyset$ satisfies the property. Thus, using the refined version of Rice’s theorem above, we can conclude $CLT_{TM}$ is not recognizable.

To show that it is not trivial: there exist a TM in $CLT_{TM}$, e.g. take $M_\emptyset$ which rejects all inputs. $\langle M_\emptyset \rangle \in CLT_{TM}$ because $\emptyset$ is a context free language. There also exists a TM not in $CLT_{TM}$, e.g take a TM $T$ that accepts all strings of the form $a^n b^n c^n$ and rejects all other strings. $\langle T \rangle$ not in $CLT_{TM}$ because $L(T) = \{a^n b^n c^n : n \geq 0\}$ is not a CFL.

To show that it’s a language property, note that for any two TMs $M_1, M_2$ with $L(M_1) = L(M_2)$, either this language is CFL, and then both $\langle M_1 \rangle, \langle M_2 \rangle \in CFL_{TM}$, or this language is not a CFL and then both $\langle M_1 \rangle, \langle M_2 \rangle \notin CFL_{TM}$. In any case, $\langle M_1 \rangle \in CFL_{TM} \iff \langle M_2 \rangle \in CFL_{TM}$.

Finally, $\emptyset$ satisfies the property since, as we already mentioned above, $\emptyset$ is a CFL.